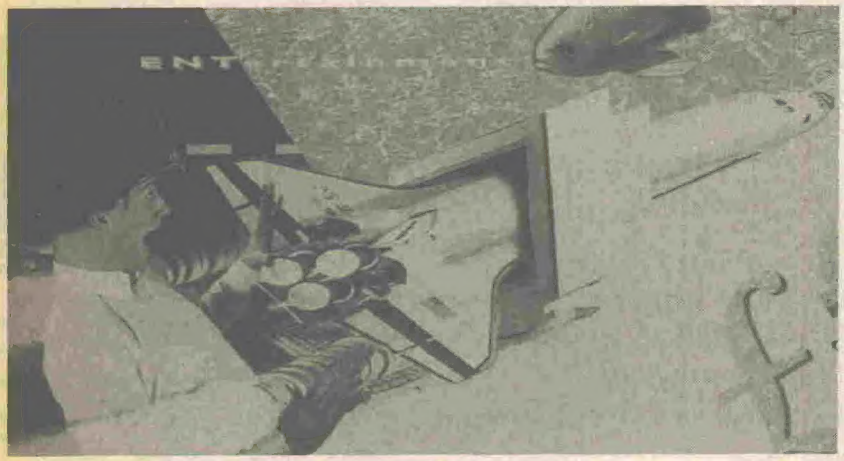
สมชัย พุทธคุณรักษา

SOFT MAIL

เทคนิคในการประมวลผลแบบติด Turbo โดยใช้อะเรย์



ใช้อะเรย์ในการประมวลผล โดยอาศัยเทคนิคการอ้างตำแหน่ง จะทำให้การคำนวณจากเวลานานนับชั่วโมง เหลือเพียงไม่ถึง 3 นาที

ในแต่ละวันผู้เขียนจะมีความสัมพันธ์กับตัวเลขและข้อมูลตัวเงินตลอดเวลา โดยมีจำนวนข้อมูลเป็นล้านรายการ โดยข้อมูลจะมีหลากหลายมาก บางครั้งต้องการคำนวณหายอดโดยการจัดกลุ่ม และแตกเป็นกลุ่มย่อย ๆ ซึ่งถ้ามีการกำหนดตัวแปรและเงื่อนไขในการประมวลผลมาก เช่น 1,000 ตัวแปร เงื่อนไขในการค้นหาและประมวลผล อาจจะต้องใช้เวลา (รอบในการประมวลผล) = (1,000*1,000,000* จำนวนบรรทัดเงื่อนไขคำสั่งประมวลผล*เวลาในการอ่าน File ในแต่ละ Record) ซึ่งจะเห็นว่ากว่าจะได้รายงานตามต้องการ อาจใช้เวลานานนับชั่วโมงเลยทีเดียว แต่ผู้เขียนใช้อะเรย์ในการประมวลผล โดยอาศัยเทคนิคการอ้างตำแหน่งจะทำให้การคำนวณจากเวลานานนับชั่วโมงเหลือเพียงไม่ถึง 2-3 นาที โดยการจับข้อมูลขึ้นมาและอ้างตำแหน่งเข้าไปรวมกลุ่มทันทีแบบรอบเดียวจบ และในบทความนี้ผู้เขียนจะขอยกตัวอย่างการอ้างตำแหน่งอะเรย์ โดยการเขียนโปรแกรมการแปลงจำนวนเงินเป็นตัวอักษรแบบสั้นกะทัดรัดและประมวลผลเร็วมาก โดย FOXPRO เพื่อเป็นแนวทางในการประยุกต์ใช้งานต่อไป

ทำความรู้จักกับอะเรย์แบบ FOXPRO

อะเรย์คือตัวแปรชุดที่เรากำหนดขึ้นเพียงชื่อเดียว และอาศัยการอ้างตำแหน่งลำดับที่เรากำหนด โดยตัวแปรชุดนั้น ๆ ข้อมูลควรเป็นข้อมูลประเภทเดียวกัน เช่น Numberic ก็ไม่ควรให้มีข้อมูลประเภท DATE หรือประเภท Charactor ปน เพราะอาจทำให้สับสนและการทำงานอาจผิดได้ง่าย ๆ

การกำหนดอะเรย์

แบบ 1 มิติ เช่น dime x(5) หมายถึง เรามีตัวแปรเกิดขึ้นแล้ว 5 ตัว คือ x(1), x(2), x(3), x(4), x(5) เราสามารถกำหนดค่าได้ดังนี้ x(1)=5, x(2)=10, x(3)=100, x(4)=200, x(5)=500 เป็นต้น การอ้างค่าเช่น

```
? x(1)+x(2)     ค่าที่ได้เท่ากับ 15
? x(5)-x(4)     ค่าที่ได้เท่ากับ 300 เป็นต้น
```

แบบ 2 มิติ เช่น dime x(5,5) หมายถึงเราจะมีตัวแปรเกิดขึ้นแล้ว 25 ตัว คือ

```
x(1,1) , x(1,2) , x(1,3) , x(1,4) , x(1,5)
x(2,1) , x(2,2) , x(2,3) , x(2,4) , x(2,5)
x(3,1) , x(3,2) , x(3,3) , x(3,4) , x(3,5)
x(4,1) , x(4,2) , x(4,3) , x(4,4) , x(4,5)
x(5,1) , x(5,2) , x(5,3) , x(5,4) , x(5,5)
```

การกำหนดค่าทำเช่นเดียวกับแบบอะเรย์ 1 มิติ แต่ต้องอ้างตำแหน่ง 2 ตำแหน่ง เช่น x(1,1)=500,

x(1,2)=100 เป็นต้น การอ้างตำแหน่งสามารถใช้ตัวแปรแทนได้ เช่น ii=1, kk=2 การแสดงผลเช่น

```
? x( ii , ii )   ค่าที่ได้เท่ากับ 500
? x( ii , kk )  ค่าที่ได้เท่ากับ 100 เป็นต้น
```

การจะเขียนโปรแกรมให้อ้างตำแหน่งนั้น จะต้องมีการวางแผนและศึกษารูปแบบของข้อมูลให้เหมาะสม การทำงานจึงจะถูกต้องและรวดเร็ว

ตัวอย่าง การแปลงจำนวนเงินเป็นตัวอักษรนั้นเราต้องกำหนดตำแหน่งอะเรย์ก่อน ตามตารางที่ 1 เพื่อนำไปสู่การสร้างอะเรย์จริง โดยเขียนเป็น program cons_mem.prg ตามลิสติงที่ 1 ค่าที่กำหนด คือ DIME TB(7,11) คือกำหนดตำแหน่ง 77 ตำแหน่ง และให้ค่าตามตาราง 1 (การสร้างตัวแปรอะเรย์ในโปรแกรมนี้สร้างเพื่อกำหนดค่าถาวรครั้งเดียว เพื่อการอ้างนำข้อมูลออกมาในภายหน้า)

TB	11	10	9	8	7	6	5	4	3	2	1
1	" "	" "	"หนึ่งล้าน"	"สองล้าน"	"สามล้าน"	"สี่ล้าน"	"ห้าล้าน"	"หกล้าน"	"เจ็ดล้าน"	"แปดล้าน"	"เก้าล้าน"
2	" "	" "	"หนึ่งแสน"	"สองแสน"	"สามแสน"	"สี่แสน"	"ห้าแสน"	"หกแสน"	"เจ็ดแสน"	"แปดแสน"	"เก้าแสน"
3	" "	" "	"หนึ่งหมื่น"	"สองหมื่น"	"สามหมื่น"	"สี่หมื่น"	"ห้าหมื่น"	"หกหมื่น"	"เจ็ดหมื่น"	"แปดหมื่น"	"เก้าหมื่น"
4	" "	" "	"หนึ่งพัน"	"สองพัน"	"สามพัน"	"สี่พัน"	"ห้าพัน"	"หกพัน"	"เจ็ดพัน"	"แปดพัน"	"เก้าพัน"
5	" "	" "	"หนึ่งร้อย"	"สองร้อย"	"สามร้อย"	"สี่ร้อย"	"ห้าร้อย"	"หกร้อย"	"เจ็ดร้อย"	"แปดร้อย"	"เก้าร้อย"
6	" "	" "	"สิบ"	"ยี่สิบ"	"สามสิบ"	"สี่สิบ"	"ห้าสิบ"	"หกสิบ"	"เจ็ดสิบ"	"แปดสิบ"	"เก้าสิบ"
7	"หนึ่ง"	" "	"เอ็ด"	"ยอง"	"สาม"	"สี่"	"ห้า"	"หก"	"เจ็ด"	"แปด"	"เก้า"

ตารางที่ 1

การอ้างหลักและตัวเลขของอะเรย์ตามตารางเพื่อนำข้อมูลมาใช้งาน

TB (หลักเงิน เช่น หลักที่ 1=ล้าน หลักที่ 2=แสน ..., ตัวเลขเงินที่จะแปลงจริง เช่น 9=เก้า 8=แปด) ตัวอย่างการอ้าง aa=7654321 หลักที่ 1=เจ็ดล้าน หลักที่ 6=หกแสน หลักที่ 3=ห้าหมื่น...หลักที่ 7=หนึ่ง ดังนั้นถ้าเราอ้างอะเรย์ในลิสติง 1 เช่น tb(1,7) ควรจะได้ตัวอักษร "เจ็ดล้าน" แต่ค่าที่ได้กลับเป็น "สามล้าน" อ้าง tb(2,6) ควรจะได้ตัวอักษร "หกแสน" แต่ค่าที่ได้ "สี่แสน" เกิดอะไรขึ้นเพราะเราไม่สามารถอ้างตำแหน่งตรงตัวได้ เพราะค่าอะเรย์ใน Foxpro ตำแหน่งที่ 0 ไม่มี ดังนั้นการอ้างตำแหน่งของ Foxpro จึงอ้างได้แค่ 1-9 เท่านั้น แล้วเลข 0 เราจะอ้างได้อย่างไรในเมื่อตัวเลขที่เรา Input นั้นมีตั้งแต่ 0-9 รวม 10 ตัว เราจึงต้องอาศัยเทคนิคในการทำให้เลข 0 นั้นไปอ้างตำแหน่งที่ 10 แทน เช่น bb=100000 เราควรจะอ้างตำแหน่ง tb(1,1), tb(2,0), tb(3,0), tb(4,0), tb(5,0), tb(6,0), tb(7,0) แต่ค่าตำแหน่ง 0 อ้างไม่ได้ ดังนั้นค่าที่ต้องอ้างจริง (ขึ้นอยู่กับการออกแบบ) คือ tb(1,9), tb(2,10), tb(3,10), tb(4,10), tb(5,10), tb(6,10), tb(7,10) เมื่อเราไปถอดค่าในตารางค่าที่ได้จึงเท่ากับ "หนึ่งล้าน" การที่จะให้เลข 0 กลายเป็น 10 ไปอ้างตำแหน่งนั้น เป็นผลทำให้ขบวนการตัวเลขนั้นต้องปรับตามไปหมด เพราะเราต้องใช้สูตรคำนวณและกำหนดให้ค่าที่ได้เป็นค่าสัมบูรณ์ (Absolute) โดย Function ดังนี้

Function abs (Expression) จะให้ค่าที่ได้ในวงเล็บเป็นค่าสัมบูรณ์ คือค่าจำนวนเต็มบวก เช่น

```
aa=0
aa=abs(aa-10) aa จะมีค่า = 10
aa=7
aa=abs(aa-10) aa จะมีค่า = 10
aa=9
aa=abs(aa-10) aa จะมีค่า = 10
```

จะเห็นว่าเมื่อเข้าสูตรค่าสัมบูรณ์แล้ว เลขที่อ้างเท่ากับ (10 ลบด้วยเลขที่อ้าง) ดังนั้นการอ้างจึงต้องอ้างตามลิสติงที่ 1 ที่ต้องออกแบบล่วงหน้าไว้ก่อนแล้ว ถ้าเป็นโปรแกรมอื่น เช่น C/C++ สามารถอ้างตำแหน่ง 0 ได้เลย การทำงานจะไม่ซับซ้อนเหมือน Foxpro กลับไปดูตัวอย่างเดิม การอ้างต้องเป็นดังนี้

ตัวอย่างการอ้าง aa=7654321 หลักที่ 1=เจ็ดล้าน หลักที่ 2=หกแสน หลักที่ 3=ห้าหมื่น.... หลักที่ 7=หนึ่ง

ค่าที่ส่ง (1,7) เรียก tb(1,abs(7-10)) การอ้างตำแหน่งจริง tb(1,3) ค่าที่ได้="เจ็ดล้าน"

ค่าที่ส่ง (2,6) เรียก tb(2,abs(6-10)) การอ้างตำแหน่งจริง tb(2,4) ค่าที่ได้="หกแสน"

ค่าที่ส่ง (3,5) เรียก tb(5,abs(5-10)) การอ้างตำแหน่งจริง tb(3,5) ค่าที่ได้="ห้าหมื่น"... เป็นต้น

ลิสติงที่ 1 Program Cons_mem.prg การสร้างอะเรย์ตัวแปรถาวร (อาจเก็บในลักษณะ File ได้)

```
PUBL TB
DIME TB(7,11)
tb(1,10)=""
tb(1,9)="หนึ่งล้าน"
tb(1,8)="สองล้าน"
tb(1,7)="สามล้าน"
tb(1,6)="สี่ล้าน"
tb(1,5)="ห้าล้าน"
tb(1,4)="หกล้าน"
tb(1,3)="เจ็ดล้าน"
tb(1,2)="แปดล้าน"
tb(1,1)="เก้าล้าน"

tb(2,10)=""
tb(2,9)="หนึ่งแสน"
tb(2,8)="สองแสน"
tb(2,7)="สามแสน"
tb(2,6)="สี่แสน"
tb(2,5)="ห้าแสน"
tb(2,4)="หกแสน"
tb(2,3)="เจ็ดแสน"
tb(2,2)="แปดแสน"
tb(2,1)="เก้าแสน"
```

→

ลิสติงที่ 1 (ต่อ)

```
tb(3,10)=""
tb(3,9)="หนึ่งหมื่น"
tb(3,8)="สองหมื่น"
tb(3,7)="สามหมื่น"
tb(3,6)="สี่หมื่น"
tb(3,5)="ห้าหมื่น"
tb(3,4)="หกหมื่น"
tb(3,3)="เจ็ดหมื่น"
tb(3,2)="แปดหมื่น"
tb(3,1)="เก้าหมื่น"

tb(4,10)=""
tb(4,9)="หนึ่งพัน"
tb(4,8)="สองพัน"
tb(4,7)="สามพัน"
tb(4,6)="สี่พัน"
tb(4,5)="ห้าพัน"
tb(4,4)="หกพัน"
tb(4,3)="เจ็ดพัน"
tb(4,2)="แปดพัน"
tb(4,1)="เก้าพัน"
tb(5,10)=""
tb(5,9)="หนึ่งร้อย"
tb(5,8)="สองร้อย"
tb(5,7)="สามร้อย"
tb(5,6)="สี่ร้อย"
tb(5,5)="ห้าร้อย"
tb(5,4)="หกร้อย"
tb(5,3)="เจ็ดร้อย"
tb(5,2)="แปดร้อย"
tb(5,1)="เก้าร้อย"

tb(6,10)=""
tb(6,9)="สิบ"
tb(6,8)="ยี่สิบ"
tb(6,7)="สามสิบ"
tb(6,6)="สี่สิบ"
tb(6,5)="ห้าสิบ"
tb(6,4)="หกสิบ"
tb(6,3)="เจ็ดสิบ"
tb(6,2)="แปดสิบ"
tb(6,1)="เก้าสิบ"

tb(7,11)="หนึ่ง"
tb(7,10)=""
tb(7,9)="เอ็ด"
tb(7,8)="สอง"
tb(7,7)="สาม"
tb(7,6)="สี่"
tb(7,5)="ห้า"
tb(7,4)="หก"
tb(7,3)="เจ็ด"
tb(7,2)="แปด"
tb(7,1)="เก้า"
retu
```

การเรียกใช้งานโปรแกรม

การเรียกใช้งานจะแสดงโปรแกรม 2 ชนิด คือ

1. แบบจำนวนเต็มบาท (mon1.prg)
2. แบบมีสตางค์ (mon2.prg)

ซึ่งจะมีข้อแตกต่างกันเล็กน้อย โดยถ้าหากมีเศษสตางค์โปรแกรมจะยาวขึ้นเล็กน้อย 2- 3 บรรทัด เนื่องจากมีการอ้างตำแหน่งอะเรย์ 2 ครั้ง คือ บาท 1 ครั้ง และสตางค์ 1 ครั้ง ดังนี้

ลิสติงที่ 2 โปรแกรมเรียกอ้างตำแหน่งแบบจำนวนเต็ม

```
***** main program mon1.prg
1.SET TALK OFF
2.SET ECHO OFF
3.do cons_mem
4.aa=0
5.dime ii(7)
6.clea
7.@ 10,1 say "จำนวนเงิน(ตัวเลข)
 : " get aa pict "9999999"
8. read
9.for jj=1 to 7
10.ii(jj)=abs(val(subs(str(aa,7),jj,1))-10)
11.endf
12.
  x1=tc(ii(1),ii(2),ii(3),ii(4),ii(5),ii(6),ii(7))
13.@ 12,1 say "ตัวอักษร
: "+iif(len(x1)<>0,x1+"บาทถ้วน"
,x1)
14.retu
```

การทำงานของโปรแกรม main หลัก ๆ

บรรทัดที่ 3 จะเรียกค่าตัวแปรอะเรย์ที่สร้างไว้ขึ้นบนหน่วยความจำ

บรรทัดที่ 5 กำหนดตัวแปรอะเรย์ 7 หลัก เพื่อเป็นตัวรองรับข้อมูลที่ input ในบรรทัดที่ 7

บรรทัดที่ 7 รับค่าตัวเงิน

บรรทัดที่ 9, 10, 11 นำตัวเลขที่รับทั้ง 7 หลักไปหาค่าสัมบูรณ์และเก็บไว้ในตัวแปรอะเรย์ ii

บรรทัดที่ 12 ส่งตัวแปรทั้งเจ็ดไปอ้างตำแหน่งอะเรย์หลักในหน่วยความจำ โดย Function TC เพื่อหาตัวอักษร

บรรทัดที่ 13 นำค่า x1 ที่ได้จากการอ้างกลับมาแสดงผล

ลิสติงที่ 3 โปรแกรมการอ้างตำแหน่งในอะเรย์

```
1.func tc
2.para a1,a2,a3,a4,a5,a6,a7
3. a7=iif(a6=10 and a7=9,11,a7)
4. zz=tb(1,a1)+tb(2,a2)+tb(3,a3)+tb(4,a4)+tb(5,a5)+tb(6,a6)+tb(7,a7)
5.retu allt(zz)
```

การทำงานหลัก ๆ ของ Function TC

บรรทัดที่ 2 รับค่าสัมบูรณ์ทั้ง 7 หลัก เพื่อเตรียมอ้างอะเรย์

บรรทัดที่ 3 ถ้าตรวจสอบว่าเลขหลักหน่วยเป็น 1 และหลักสิบเป็น 0 ค่าที่เรียกหลักหน่วยคือ "หนึ่ง" ให้กำหนดการอ้างไปที่ tb(7,11) คือ "หนึ่ง" ถ้าหลักสิบมีค่าไม่เท่ากับ 0 และหลักหน่วยเป็น 1 ต้องอ่านเป็น "เอ็ด" ให้อ้างปกติคือ tb(7,9)

บรรทัดที่ 4 อ้างอะเรย์ในหน่วยความจำแล้วนำค่าที่ได้ใส่ตัวแปร zz

บรรทัดที่ 5 ส่งค่าที่ได้จากการอ้างตำแหน่งคืนให้โปรแกรม Main

ลิสติงที่ 4 โปรแกรมเรียกอ้างตำแหน่งแบบมีเศษสตางค์

```
***** main program mon2.prg
1. SET TALK OFF
2. SET ECHO OFF
3. do cons_mem
4. aa=0.00
5. dime ii(7)
6. clea
7. @ 10,1 say"จำนวนเงิน(ตัวเลข)
 : " get aa pict "9999999.99"
8. read
9. for jj=1 to 7
10. ii(jj)=abs(val(subs(str(aa,10,2),jj,1))-10)
11. endf
12. x1=tc(ii(1),ii(2),ii(3),ii(4),ii(5),ii(6),ii(7))
13. x2=tc(10,10,10,10,10,abs(val(subs(str(aa,10,2),9,1))-
10),abs(val(subs(str(aa,10,2),10,1))-10))
14.x1=iif(len(x2)=0,x1+"บาทถ้วน",x1+"บาท"+x2+"สตางค์")
15. @ 12,1 say "ตัวอักษร : "+iif(left(x1,3)="บาท","",x1)
16.retu
```

การทำงานของโปรแกรม main หลัก ๆ

บรรทัดที่ 3 จะเรียกค่าตัวแปรอะเรย์ที่สร้างไว้ขึ้นบนหน่วยความจำ

บรรทัดที่ 5 กำหนดตัวแปรอะเรย์ 7 หลัก เพื่อเป็นตัวรองรับข้อมูลที่ input ในบรรทัดที่ 7

บรรทัดที่ 7 รับค่าตัวเงิน

บรรทัดที่ 9, 10, 11 นำตัวเลขที่รับทั้ง 7 หลัก ไปหาค่าสัมบูรณ์และเก็บไว้ในตัวแปรอะเรย์ ii

บรรทัดที่ 12 ส่งตัวแปรทั้งเจ็ดไปอ้างตำแหน่งอะเรย์หลักในหน่วยความจำ โดย Function TC เพื่อหาตัวอักษรจำนวนเต็มบาท

บรรทัดที่ 13 ส่งค่าสัมบูรณ์หลักที่ 9,10 คือสตางค์ ไปอ้างอะเรย์หลักในหน่วยความจำ โดย Function TC เพื่อหาสตางค์ โดยส่งค่าหลอก 5 ค่า คือ เลข 10 ไปด้วย (เลข 10 เมื่อไปอ้างจะได้ความว่างเปล่ากลับมา)

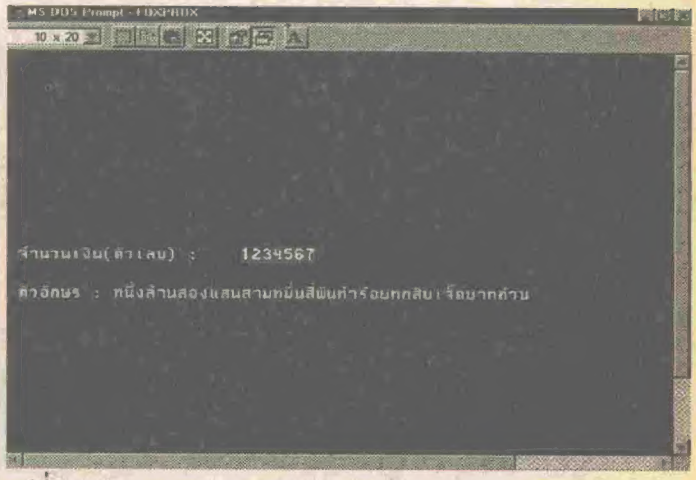
บรรทัดที่ 14 นำค่า x1 คือบาท และ ค่า x2 คือสตางค์มาต่อกันที่ได้จากการอ้างอะเรย์

บรรทัดที่ 15 นำค่า x1 ที่ได้มาแสดงผล

ลิสติงที่ 5 โปรแกรมการอ้างตำแหน่งในอะเรย์

```
func tc
para a1,a2,a3,a4,a5,a6,a7
  a7=iif(a6=10 and a7=9,11,a7)
  zz=tb(1,a1)+tb(2,a2)+tb(3,a3)+tb(4,a4)+tb(5,a5)+tb(6,a6)+tb(7,a7)
retu allt(zz)
```

ผลการรันแบบจำนวนเต็มจะได้ตามรูปที่ 1 และแบบมีสตางค์จะได้ตามรูปที่ 2



รูปที่ 1

รูปที่ 2

สรุปส่งท้าย

การใช้อะเรย์ในการประมวลผลนั้นมีความเร็วสูง และโปรแกรมที่เขียนก็สั้นลง ซึ่งเราสามารถดัดแปลงนำไปใช้งานได้สารพัด จากตัวอย่างข้างต้น อะเรย์ที่เรากำหนดจาก Cons_mem.prg อาจจะ Save อะเรย์ที่ได้ในลักษณะ File และหากต้องการใช้งานก็ Restore ขึ้นมาใช้งานโดยไม่ต้องสร้างอะเรย์ทุกครั้งก็ได้ แล้วแต่เทคนิคแต่ละบุคคล และหรือจะประยุกต์ให้สามารถเรียกอ้างตำแหน่งเงินให้มากขึ้นได้ไม่จำกัดหลัก โดยเพิ่มเงื่อนไขในลักษณะการหาสตางค์ โดยการส่งค่าครั้งละ 7 หลัก แล้วนำค่าที่ได้มาต่อกัน การออกแบบและการเขียนโปรแกรมผู้เขียนบทความนี้จะพิจารณาว่าหากงานที่ทำต้องการความเร็ว ขั้นแรกต้องพิจารณาอะเรย์ก่อน แต่ถ้าเป็นโปรแกรมประเภท Compiler เช่น C/C++ นั้น ต้องกำหนดค่าอะเรย์ตายตัว (บางครั้งมาก-น้อยเกินความต้องการ) ก่อนคอมไพล์ ซึ่งหากเราต้องการความยืดหยุ่นของตัวแปรเพราะขอบเขตของข้อมูลบางครั้งไม่แน่นอน เราก็สามารถใช้ File แทนอะเรย์หรือตัวแปรได้เช่นกัน หากมีโอกาสจะเขียนเล่านำเสนอต่อไป หวังเป็นอย่างยิ่งว่า คงจะเป็นแนวทางเพื่อนำไปประยุกต์ใช้งานของผู้อ่านได้บ้าง สวัสดีครับ